AFRL-AFOSR-VA-TR-2017-0075

Stochastic Hybrid Systems Modeling and Middleware-enabled DDDAS for Next-generation US Air Force Systems

Aniruddha Gokhale
VANDERBILT UNIVERSITY
110 21ST AVENUE S STE 937
NASHVILLE, TN 37203-2416

03/30/2017
Final Report

DISTRIBUTION A: Distribution approved for public release.

Air Force Research Laboratory
AF Office Of Scientific Research (AFOSR)/RTA2

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
30-03-2017

2. REPORT TYPE
Final Performance

3. DATES COVERED *(From - To)*
30 Sep 2013 to 31 Dec 2016

4. TITLE AND SUBTITLE
Stochastic Hybrid Systems Modeling and Middleware-enabled DDDAS for Next-generation US Air Force Systems

5a. CONTRACT NUMBER

5b. GRANT NUMBER
FA9550-13-1-0227

5c. PROGRAM ELEMENT NUMBER
61102F

6. AUTHOR(S)
Aniruddha Gokhale, Douglas Schmidt, Xenofon Koutsoukos

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
VANDERBILT UNIVERSITY
110 21ST AVENUE S STE 937
NASHVILLE, TN 37203-2416 US

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AF Office of Scientific Research
875 N. Randolph St. Room 3112
Arlington, VA 22203

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/AFOSR RTA2

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-AFOSR-VA-TR-2017-0075

12. DISTRIBUTION/AVAILABILITY STATEMENT
DISTRIBUTION A: Distribution approved for public release.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This document presents the final report on the research investigations and outcomes of our AFOSR DDDAS project titled Stochastic Hybrid Systems Modeling and Middleware-enabled DDDAS for Next-generation US Air Force Systems. It summarizes our contributions to the various facets of the DDDAS paradigm when applied to provide dynamic resource management in cloud computing platforms so that they can support applications with different quality of service requirements. To that end, first, we describe our approach on workload characterization of cloud-hosted applications using online model learning that is used for resource management in the cloud. Second, we report on a new service called Simulation-based Optimization as a Service, which is an approach we have developed to simulate the learned models to obtain optimal values of parameters to a model that are applied to the system in the DDDAS feedback loop. Third, we report on a number of dynamic resource management algorithms we have developed and their experimental evaluations for hosting DDDAS-like applications in public cloud infrastructures. Finally, we report on ongoing work towards using the DDDAS paradigm in the continuum from cloud to the edge to support applications that are hosted across the

15. SUBJECT TERMS
complex systems autonomy stochastic hybrid systems

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			LEVE, FREDERICK
Unclassified	Unclassified	Unclassified	UU		

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

DISTRIBUTION A: Distribution approved for public release.

					19b. TELEPHONE NUMBER *(Include area code)* 703-696-7309

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

DISTRIBUTION A: Distribution approved for public release.

Stochastic Hybrid Systems Modeling and Middleware-enabled DDDAS for Next-generation US Air Force Systems

Final Report for AFOSR DDDAS FA9550-13-1-0227, March 2017

PIs: Aniruddha Gokhale, Xenofon Koutsoukos and Douglas C. Schmidt
Students: Faruk Caglar, Shashank Shekhar, Hamzah Abdel Aziz, Shweta Khare, and Michael Walker
Department of Electrical Engineering and Computer Science
Vanderbilt University, Nashville, TN 37235, USA
Contact Email of lead PI: a.gokhale@vanderbilt.edu

Abstract—**This document presents the final report on the research investigations and outcomes of our AFOSR DDDAS project titled Stochastic Hybrid Systems Modeling and Middleware-enabled DDDAS for Next-generation US Air Force Systems. It summarizes our contributions to the various facets of the DDDAS paradigm when applied to provide dynamic resource management in cloud computing platforms so that they can support applications with different quality of service requirements. To that end, first, we describe our approach on workload characterization of cloud-hosted applications using online model learning that is used for resource management in the cloud. Second, we report on a new service called Simulation-based Optimization as a Service, which is an approach we have developed to simulate the learned models to obtain optimal values of parameters to a model that are applied to the system in the DDDAS feedback loop. Third, we report on a number of dynamic resource management algorithms we have developed and their experimental evaluations for hosting DDDAS-like applications in public cloud infrastructures. Finally, we report on ongoing work towards using the DDDAS paradigm in the continuum from cloud to the edge to support applications that are hosted across the cloud-edge spectrum.**

Keywords-**Dynamic resource management, model learning, simulation-based optimizations, cloud infrastructures for DDDAS applications.**

I. Introduction

Critical cyber-physical infrastructure, such as the national power grid, transportation network [1] and smart cities [2], are large-scale and complex systems that illustrate highly dynamic and uncertain nature of the operations, as well as significant heterogeneity in the end systems, network protocols and technologies, and software systems that support the system operations. In such systems, human intervention becomes infeasible to handle problems stemming from cyber-physical events such as failures or deliberate attacks.

Further, it is estimated that with increasing mobility, the mobile traffic will have grown thirteen times more than the existing mobile traffic and there will be three times more connected devices than the number of people on the Earth [3]. Similarly, scientific experiments such as CERN also generate enormous amounts of data estimated to be about twenty-five petabytes in a year [4]. With the emergence of the Internet of Things (IoT) paradigm, billions of data points are generated and as a result, the volume of this data is getting even larger.

All of this generated data must be processed to extract useful features out of it. This growing, massive amounts of data require more storage and compute resources, which is ultimately provided by the data centers throughout the world and the cloud computing infrastructure. As more and more applications are created, the cloud computing in general and data center in particular have become critical for many projects, enterprises, and research communities. Hence, it will continue to play a crucial role in delivering a variety of services. Many of these services will require a variety of quality of service (QoS) properties to be supported, which means that the cloud platforms must provide differentiated services to different cloud-hosted applications, in turn requiring effective resource management solutions for the shared cloud platforms.

Despite the fact that there is a significant momentum towards moving to the cloud, a variety of issues still exist in utilizing the cloud to its fullest potential. For example, energy efficiency, capacity planning, performance management, disaster management, and security are a few major concerns faced by cloud service providers (CSPs) among others. The energy consumption of data centers worldwide has reached staggering proportions and this trend will further continue. Moreover, diesel power generators, due to power outages in data centers and power plants, emit millions of tons of carbon [5], [6]. Thus, CSPs must address energy efficiency issues for data centers. A recent initiative by the US Department of Energy (DOE) seeks the data centers to become 20% more energy efficient by 2020 [7].

A. Solution Approach: Use the DDDAS Principles

The *Dynamic Data Driven Applications Systems* (DDDAS) [8] principles are a promising approach to address the need to manage and control the next generation of cloud-hosted cyber-physical systems. DDDAS prescribes a data-driven model learning process of real-world systems and subsequently simulating these models within a decision support system to control the system behavior and maintaining its intended trajectory. The use of simulations in decision support is fundamental as a means to enable dynamic data-driven decision support in a wide array of systems. However, the success of any DDDAS approach depends on its ability to learn and simulate models of the target system. In turn, the quality of the learned models will determine how effectively the real-world system can be managed and controlled.

Figure 1 illustrates why the cloud resource management is an important problem for our project. We envision a variety of DDDAS applications with their QoS needs will be executing on public cloud infrastructures. To satisfy the timeliness and reliability needs of the cloud-hosted DDDAS applications, we must assure these properties at the cloud level through effective resource management strategies.

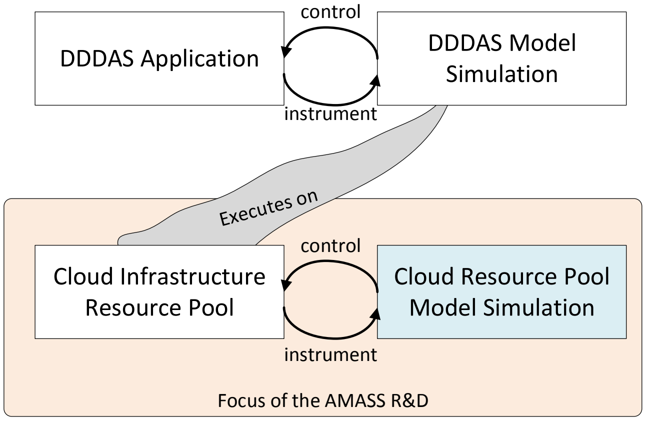


Fig. 1. AMASS Project Focus

To that end, the cloud platform must be managed and controlled so that it can provide the desired functional and non-functional properties of the cloud-hosted DDDAS applications. In order to accomplish this objective, the cloud platform itself can be viewed as yet another DDDAS system, which means that the models of the cloud platform must be learned and simulated to enforce the right resource management decisions. This key philosophy defined our contributions for the three years of the project. In this project, we made the following contributions each of which is summarized in this report.

1) We describe our approach to workload characterization of cloud-hosted applications as a way to describe a strategy for online model learning that can be used for resource management in the cloud.
2) We describe a new service called Simulation-based Optimization as a Service, which is an approach we have developed to simulate the learned models to obtain optimal parameter values of the models that are applied to the system in the DDDAS feedback loop.
3) We report on a number of dynamic resource management algorithms we have developed and their experimental evaluations for hosting DDDAS-like applications in public cloud infrastructures.
4) We report on ongoing work towards using the DDDAS paradigm in the continuum from cloud to the edge to support applications that are hosted across the cloud-edge spectrum.

B. Report Organization

The rest of the report is organized as follows: Section II details the key technical challenges we have addressed to date in our work; Section III describes our approach to model learning using Gaussian Process; Section IV describes our approach to model execution using simulation-based optimizations; Section V describes our approach to scaling dataflow programming models; Section VI outlines the different resource management solutions we have developed to date; Section VII alludes to ongoing and proposed work; and finally Section VIII offers concluding remarks summarizing our work, outcomes and alluding to ongoing work.

II. Research Challenges for Cloud Data Centers

This section provides an overview of the research challenges stemming from supporting the execution and quality of service demands of DDDAS application models executin in cloud infrastructures that are addressed by our project. To better situate these challenges, a high-level architecture of a cloud data center, which provides the resources, is depicted in Figure 2. In this architecture, physical resources such as servers are part of the physical layer which are virtualized by the virtual machine manager (VMM) or the so-called hypervisor in the virtualization layer.Lightweight forms of virtualization offered by containers is also part of this layer. Virtualized resources and infrastructure are controlled by the infrastructure management tools in the cloud management layer. Applications and cloud services are executed within the virtualized resources shown on top of the virtualization layer.

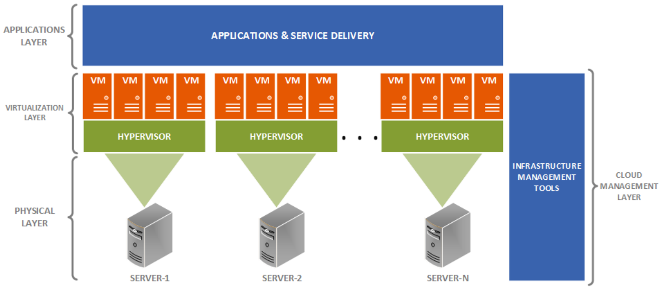


Fig. 2. High-level Architecture of a Cloud Data Center

A. Challenge 1: Model Learning Challenges

We have observed that cloud service providers use virtualization to enable hosting multiple applications in a single server such that each application has its own configuration and allocated resources to fulfill their application-specific demands and requirements. Clouds are important for model learning and execution of critical resources due to their elastic nature. Furthermore, service providers use resource overbooking to increase the utilization of the servers and therefore to increase their profit. However, the primary drawback in resource overbooking is performance interference between the hosted virtual machine (VM) collocated in the same physical host. Performance interference significantly affects application performance and its other quality-of-service (QoS) properties.

The performance interference level depends on the type of collocated workloads and their corresponding collocated resources. For example, collocating multiple VMs all with memory intensive workloads in the same physical platform (host) can lead to a high cache miss ratio rate because of their high demand for memory access. Thus, the performance interference can be minimized by introducing a smart placement or migration strategy for VMs based on their workload types and how does it impact the system performance metrics.

B. Challenge 2: Model Execution Challenges for Effective Decision Support

With the advent of data-acquisition technology in the past decade, using simulation-based optimizations provides a low cost alternative to emulation of physical phenomena including stochastic processes and solving optimal control problems in dynamic systems as has been demonstrated in many industrial applications [9], [10], [11], [12]. To provide high quality decision support, one can use simulations in an optimization loop to derive good values of system parameters for a given system state, particularly when the system has too many parameters and traditional means to optimize the outcomes are either intractable or infeasible (for example, if gradient information is not available or is hard to compute). To that end, simulation-based optimization methods have emerged to enable optimization in the context of complex, black-box simulations obviating the need for specific and accurate model information, such as gradient computation.

Despite this promise, the traditional simulation-based approaches without dynamic data driven capabilities are not able to synchronize with real-world conditions, which often results in inaccurate prediction and failure of the system control. To that end, DDDAS, as an innovative paradigm for real-time computer simulations, effectively overcomes setbacks in traditional simulation approaches. Two key challenges emerge in this context. First, although simulation based optimization has become an important subject in various areas, to solve large scale problems, simulations sometimes are extremely complex and require tremendous computing power. Second, even with DDDAS as an enabling paradigm, simulation-based optimization methods are not intended for anytime use, and do not account for real-time constraints and associated trade-offs between solution quality and time to decision, which may be critical considerations for the systems that utilize these approaches for control.

C. Challenge 3: Parallel Dataflow Models for Real-time Decision Support

With increasing importance of IoT, which is a significant expansion of the Internet to include physical devices, bridging the divide between the physical world and cyberspace becomes critical. DDDAS-centric IoT, also called Industrial IoT (IIoT), which is distinct from consumer IoT, will help realize critical infrastructures, such as smart-grids, intelligent transportation systems, advanced manufacturing, health-care tele-monitoring, etc. They share several key cross-cutting aspects. First, they are often large-scale, distributed systems comprising several, potentially mobile, publishers of information that produce large volumes of asynchronous events. Second, the resulting unbounded asynchronous streams of data must be combined with one-another and with historical data and analyzed in a responsive manner. While doing so, the distributed set of resources and inherent parallelism in the system must be effectively utilized. Third, the analyzed information must be transmitted downstream to a heterogeneous set of subscribers. In essence, the emerging IIoT systems can be understood as a *distributed asynchronous dataflow*. The key challenge lies in developing a dataflow-oriented programming model and a middleware technology that can address both *distribution* and *asynchronous processing* requirements adequately.

The distribution aspects of dataflow-oriented systems can be handled sufficiently by data-centric publish/subscribe (pub/sub) technologies [13], such as Object Management Group (OMG)'s Data Distribution Service (DDS) [14]. DDS is an event-driven publish-subscribe middleware that promotes asynchrony and loose-coupling between data publishers and subscribers which are decoupled with respect to (1) *time* (*i.e.*, they need not be present at the same time), (2) *space* (*i.e.*,they may be located anywhere), (3) *flow* (*i.e.*, data publishers must offer equivalent or better quality-of-service (QoS) than required by data subscribers), (4) *behavior* (*i.e.*, business logic independent), (5) platforms, and (6) programming languages. In fact, as specified by the Reactive Manifesto [15], event-driven design is a pre-requisite for building systems that are *reactive*,*i.e.* readily responsive to incoming data, user interaction events, failures and load variations- traits which are desirable of critical IIoT systems. Moreover, asynchronous event-based architectures unify scaling up (e.g., via multiple cores) and scaling out (e.g., via distributed compute nodes) while deferring the choice of the scalability mechanism at deployment-time without hiding *the network* from the programming model. Hence, the asynchronous and event-driven programming model offered by DDS makes it particularly well-suited for demanding IIoT systems.

However, the data processing aspects, which are local to the individual stages of a distributed dataflow, are often not implemented as a dataflow due to lack of sufficient composability and generality in the application programming interface (API) of the pub/sub middleware. DDS offers various ways to receive data such as, listener callbacks for push-based notification, read/take functions for polling, waitset and read-condition to receive data from several entities at a time, and query-conditions to enable application-specific filtering and demultiplexing. These primitives, however, are designed for data and meta-data *delivery* as opposed to *processing*. Further, the lack of proper abstractions forces programmers to develop event-driven applications using the observer pattern–disadvantages of which are well documented [16].

D. Challenge 4: Resource Management

A number of resource management challenges exist in managing cloud data centers.

1) Challenge 4a: Autonomous and Dynamic Scheduler Reconfiguration: At the virtualization layer of a data center, hypervisors have a scheduling mechanism to deal with sharing CPU resources among the virtual machines (VMs) and executing the workloads in the VMs. Borrowed Virtual Time (BVT), Simple Earliest Deadline First (sEDF), Credit, and the ESX / ESXi scheduler are a few examples of the schedulers employed by virtual machine managers. Since these schedulers are applicable to many environments and application needs, they are designed to be highly configurable where the chosen parameters for these configurations define how the VMs will be handled and orchestrated, and ultimately the performance delivered to applications hosted in the VMs.

Relying on default values, manually tuning the scheduler's parameters by following known configuration patterns, using generally accepted rules, and adopting trial-and-error approach, are common practices among the system administrators of the cloud data center. However, these approaches are not effective and efficient, particularly when dealing with dynamically changing workloads on the host machines and varied CPU resource utilizations. Moreover, these non-scientific approaches do not consider the resource overbooking ratios for resource management. Furthermore, often these manual decisions are made offline, which invariably cannot consider the overall system dynamics leading to poor system performance. Therefore, an online, autonomous, and self-tuning system for scheduler configuration is desired.

2) Challenge 4b: Resource-Overbooking to Support Soft Real-time Applications: Under-utilization, wastage of resources, and inefficient energy consumption are among the traditional issues of crucial importance to data centers. The tools in the cloud management layer in a data center are required to monitor, provision, optimize, and orchestrate the underlying cloud infrastructure resources to remedy these issues. CSPs often overbook their resources by utilizing the tools in the cloud management layer. Overbooking is an attractive strategy to CSPs because it helps to reduce energy consumption and increase resource utilization in the data center by packing more user jobs in a fewer number of resources while improving their profits. Overbooking becomes feasible because cloud users tend to overestimate their resource requirements, utilizing only a fraction of the allocated resources. Without overbooking, resources in a data center will otherwise remain under-utilized.

One common way for the data center vendors to overbook resources is to have a pre-determined one-size-fits-all overbooking ratio or a method that will determine the ratio of resource overbooking. Resource overbooking ratios are generally determined sporadically by analyzing the historic resource usage of workloads or following the best practices. Unfortunately, governing cloud resources in this manner may be detrimental and catastrophic to soft real-time applications running in the cloud. To make systematic and online determination of overbooking ratios such that the quality of service needs of soft real-time systems can be met while still benefiting from overbooking, there is a need for more efficient, effective, and intelligent approaches to overbooking that will ensure good performance for soft real-time applications yet prevent under utilization and also save energy costs.

3) Challenge 4c: Performance Interference Effects on Application Performance: Recall that it is a standard practice for CSPs to overbook physical system resources to maximize the resource utilization and make their business model more profitable. Resource overbooking is usually achieved through the tools in the cloud management layer. However, resource overbooking can lead to performance interference and anomalies among the VMs hosted on the physical resources, causing performance unpredictability for soft real-time applications hosted in the VMs. Such unpredictability may be detrimental to the performance of the DDDAS applications that are controlled by the models executing in the cloud infrastructure. Moreover, resource overbooking can propagate and trigger faults in other VMs, which is also not acceptable to DDDAS applications. To address these problems and because workloads of the VMs may change at run time, virtual machine migration between physical host machines and data centers is the generally accepted mechanism.

Choosing the right set of target physical host machines for VM migration decisions plays a critical role in determining the performance and interference effects post migration. Analyzing the performance anomalies that might occur and predicting performance interference and fault before a VM is deployed or migrated on the physical host machines is thus desired and vital for soft-real time applications.

4) Challenge 4d: Power- and Performance-Aware Virtual Machine Placement: As mentioned above, virtual machines are migrated from one physical host machine to another one in the same data center or across the data centers located in different locations due to fault tolerance, balance workload, application performance management concerns, and eliminate hotspots. Deploying, handling, and migrating VMs in a data center are managed by the tools in cloud management layer.

Apart from the performance interference aspects described above, power and performance trade-offs are also critical and challenging issues faced by CSPs while managing their data centers. On the one hand, CSPs strive to reduce power consumption of their data centers to not only decrease their energy costs but also to reduce adverse impact on the environment. On the other hand, CSPs must deliver performance expected by the applications hosted in their cloud data centers in accordance with predefined Service Level Objective (SLOs). Not doing so will lead to loss of customers and thereby major revenue losses for the CSPs. Power management and performance assurance are conflicting objectives, particularly in the context of multi-tenant cloud systems where multiple VMs may be hosted on a single physical server. The problem becomes even harder when soft real-time applications are hosted in these VMs.

Solutions to address the virtual machine placement decisions exist. Bin packing heuristics such as first-fit, best-fit, and next-fit are common practices used by cloud management platforms (e.g., OpenNebula, OpenStack, etc.) to deploy VMs in the cloud. However, these solutions do not consider application performance and energy efficiency. To address the aforementioned issues, a power and performance-aware virtual machine placement algorithm is desired.

5) Challenge 4e: Supporting Stochastic Hybrid Models of DDDAS Applications: With the advent of the Internet of Things (IoT) paradigm [17], which involves the ubiquitous presence of sensors, there is no dearth of collected data. When coupled with technology advances in mobile computing and edge devices, users are expecting newer and different kinds of services that will help them in their daily lives. For example, users may want to determine appropriate temperature settings for their homes such that their energy consumption and energy bills are kept low yet they have comfortable conditions in their homes. Other examples include estimating traffic congestion in a specific part of a city on a special events day. Any service meant to find answers to these questions will very likely require substantial number of computing resources. Moreover, users will expect a sufficiently low response time from the services.

Deploying these services in-house is unrealistic for the users since the models of these systems are quite complex to develop. Some models may be stochastic in nature, which require a large number of compute-intensive executions of the models to obtain outcomes that are within a desired statistical confidence interval. Other kinds of simulation models require running a large number of simulation instances with different parameters. Irrespective of the simulation model, individual users and even small businesses cannot be expected to acquire the needed resources in-house. Cloud computing then becomes an attractive option to host such services particularly because hosting high performance and real-time applications in the cloud is gaining traction [18], [19]. Examples include soft real-time applications such as online video streaming (e.g., Netflix hosted in Amazon EC2), gaming (Microsoft's Xbox One and Sony's Playstation Now) and telecommunication management [20].

Given these trends, it is important to understand the challenges in hosting such simulations in the cloud. To that end we surveyed prior efforts [21], [22], [23], [24] that focused on deploying parallel discrete event simulations (PDES) [25] in the cloud, which reveal that the performance of the simulation deteriorates as the size of the cluster distributed across the cloud increases. This occurs due primarily to the limited bandwidth and overhead of the time synchronization protocols needed in the cloud [26]. Thus, cloud deployment for this category of simulations is still limited.

Despite these insights, we surmise that there is another category of simulations that can still benefit from cloud computing. For example, complex system simulations that require statistical validation or those that compare simulation results under different constraints and parameter values often need to run repeatedly are suited to cloud hosting. Running these simulations sequentially is not a viable option as user expectations in terms of response times have to be met. Hence there is a need for a simulation platform where a large number of independent simulation instances can be executed in parallel and the number of such simulations can vary elastically to satisfy specified confidence intervals for the results. Cloud computing becomes an attractive platform to host such capabilities [27]. To that end we have architected a cloud-based solution comprising resource management algorithms and middleware called Simulation-as-a-Service (SIMaaS).

III. Model Learning for Performance Management

This section illustrates our DDDAS approach we use to overcome the Challenge 1 in order to support the system execution and quality of service demands. Our DDDAS architecture targets DDDAS systems whose dynamics depend on uncontrolled input along with a controlled input as shown in Figure 3. Details of our approach has been submitted to a special issue of Springer Cluster Computing on DDDAS that PI Gokhale is guest co-editing with other DDDAS PIs [28].

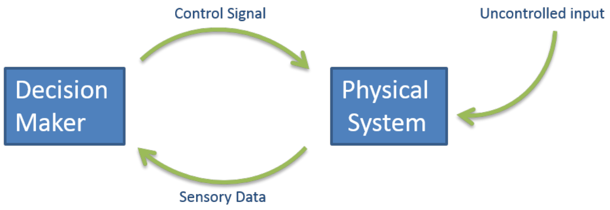


Fig. 3. DDDAS system

To model this system, we use stochastic hybrid systems (SHS) to abstract the system behavior. SHS modeling paradigm allows us to model systems which incorporate continuous nonlinear dynamics, multiple discrete modes of operations, and uncertainty. Also, we utilize advanced machine learning techniques to support our modeling paradigm with online learning capability in order to autonomously adapt our system model with the variability in the system behavior and to elevate the decision intelligence of the system decision maker. To do so, our DDDAS architecture performs three main tasks iteratively: model learning, short-term prediction and control as shown in Figure 4.

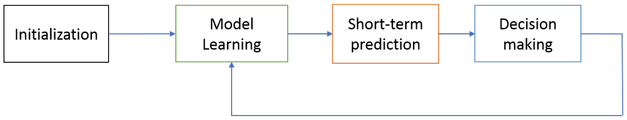


Fig. 4. DDDAS Architecture for Online learning

To formalize our system model, let us define $Y_k \in \mathbb{R}$ as the system performance (e.g. webserver Latency) at time k, $\mathbf{x}_k \in \mathbb{R}^D$ as the system state at time k (e.g. system's resource utilization), $w_k \in \mathbb{R}$ as the uncontrolled input (e.g. workload input) of our system, and $m_k \in [1 : M]$ is the mode in which the system operate. Therefore,

$$Y_{k+1} = \begin{cases} f_1(\mathbf{x}_k, w_k), & \text{if } m_k = 1 \\ & \vdots \\ f_M(\mathbf{x}_k, w_k), & \text{if } m_k = M \end{cases}$$

Also, let us define the collected data of size N as $\mathcal{D} = \{(\mathbf{x}_i, w_i, y_i) | i = 1, ..., N\}$. Our objective is to learn these model $f_1, f_2, ...f_M$ from the available data $\mathcal{D}$.

Model learning process consists of learning three type of models: clustering model, continuous nonlinear models, and time-series model. The clustering model and continuous models is used to build a SHS for the performance model Y_k. The time-series model is used to model the uncontrolled input signal (workload model) as a function of time $w_k \sim f_w(t)$ which allows us to forecast the workload input. Therefore, our model learning algorithm starts by clustering the collected data $\mathcal{D}$ using a K-mean algorithm in order to identify the system's modes of operation and to segment the data based on its corresponding mode of operation $\mathcal{D}_i, i = 1, \cdots, M$. Also, we uses Silhouette scoring to identify the number of system's modes M which best fit the data. After clustering the data, each segment of the data $\mathcal{D}_i$ is used to learn a stochastic nonlinear model which abstract the performance behavior of the system in this mode. We uses an independent Gaussian Process to learn the system performance model for each mode such that $f_i(\mathbf{x}_k, w_k) \sim \mathcal{GP}_i(m_i(\hat{\mathbf{x}}), k_i(\hat{\mathbf{x}}, \hat{\mathbf{x}}))$ where $\hat{\mathbf{x}}$ is defined as the tuples $(\mathbf{x}_k, w_k)$. Lastly, we build a time-series model using an additional Gaussian Process to model the workload input $f_w(t) \sim \mathcal{GP}(m(t), k(t, t))$. We repeat this learning process each time we receive a new data.

After learning the performance models, we perform a short-term prediction in order to estimate the system performance (i.e. $p(Y_{k+1})$) and generate the control signal accordingly. First, we identify the current system mode by classifying the current state of the system. This classification allow us to determines which performance model to use for prediction. Also, we forecast the uncontrolled input where we use the predicted mean $\bar{w}_k$ to predict the system performance.

Based on the predicted distribution of the system performance (i.e. $p(Y_{k+1})$), we generate the control signal whether to scale the VM resources up or down as shown in Figure 5.

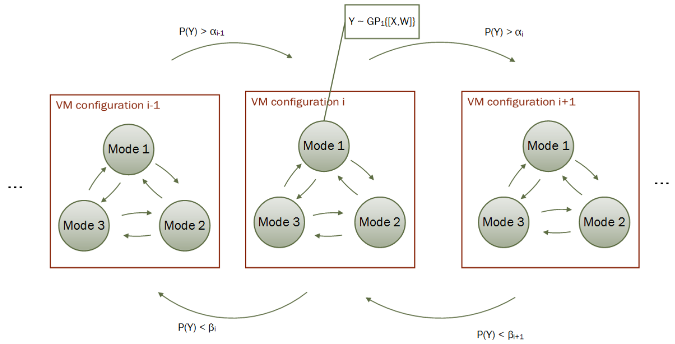


Fig. 5. System model with control

Algorithm 1: Online Learning of Performance Models

```
initilization:
while TRUE do
    Update the system Model:
    Update the forecast Model:
    Forecast the workload:
    Predict the system performance (latency):
    Scale the VM recurses:
```

Gaussian Process (GP) is a non-parametric model which uses the observed data to model the system behavior [29]. A GP is identified by its mean and covariance functions. The mean function represents the expected value before observing any data and the covariance function (also called kernel) identifies the expected correlation between the observed data.

IV. Simulation-based Optimization-as-a-Service

In this section we use a motivational case study to develop the problem statement we have formulated and solved as our approach to address Challenge 2. Our aim in this report is to provide the high level idea. The details of the approach are currently in submission to the First Annual Conference on Dynamic Data Driven Applications Systems [30].

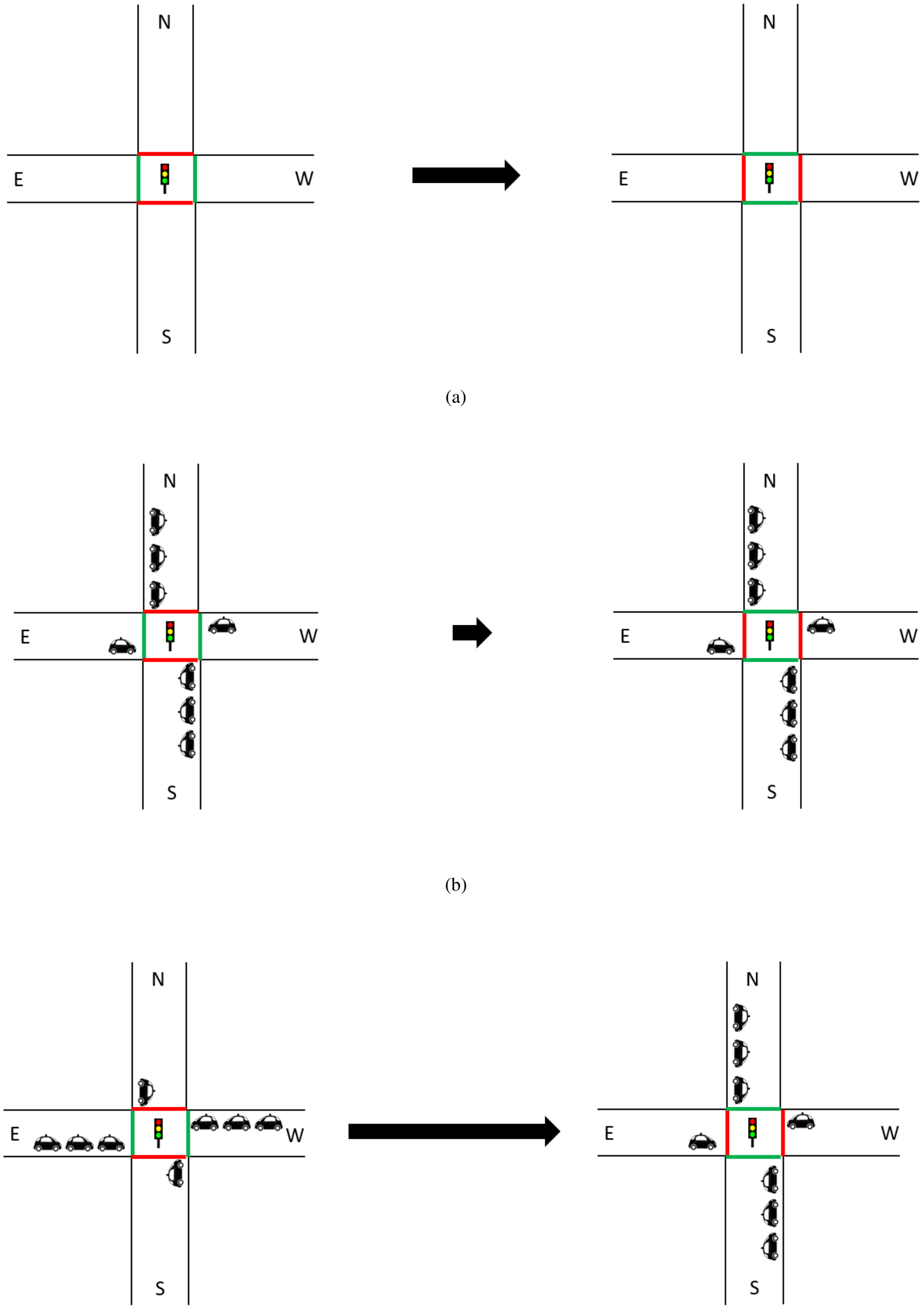
To concretely present our idea, we first present a traffic light control system as an example of a real-world system where high-quality configuration of the traffic light controller requires an iterative black-box optimization process based on data-driven model simulations. Owing to the high demand for resources and real time performance constraints, such a capability requires cloud computing resources. We designed and implemented SBOaaS, a framework for simulation-based optimization as a service. This section presents key features and a case study illustrating those challenges that SBOaaS should address.

A. Motivating Case Study: Dynamic Traffic Light Control System

To formulate the problem statement, we use a dynamic traffic light control scenario as our motivating example. In this scenario, each intersection traffic light controller switches its traffic light phases according to the observed vehicle flow. In general, a traffic light phase is related to a collection of lanes dominated by such a phase; if the number of waiting vehicles in the lanes related to the current phase is small and the number of waiting vehicles in the lanes related to the next phase is large, the controller will switch the traffic light phase. Figure 6 provides a visual demonstration of the controller logic.

Formally, a feedback controller has a predefined phase sequence $(p_0, ..., p_n)$. For each phase p_i, m_i is the minimum interval, M_i is the maximal interval, q_i is the average queue length of the lanes related to the i^{th} phase, and θ_i is the threshold on the queue length of lanes blocked in the i^{th} phase.

The controller must solve an optimization problem as follows: for a given vehicle flow of an area in a certain time



(a)

(b)

(c)

Fig. 6. The control logic for feedback controllers. (a) Non-feedback controllers have a fixed interval between two phases. (b,c) Feedback controllers dynamically change the interval according to the length of their vehicle queues.

period and a set of controlled intersections $I\{I_0,...,I_m\}$, find the optimal thresholds $(\Theta_0,...,\Theta_m)$, where $\Theta_i = (\theta_0,...\theta_{n_i})$ are the thresholds of the ith intersection.

The scenario with a single intersection with similar control logic has been discussed in many prior research efforts, e.g., [31]. However, the situation becomes much more complicated when generalizing the controller model to cases with multiple intersections and correspondingly multiple traffic lights. Many factors, such as densities of vehicle flows and topological structures of road networks, may affect the outcomes of such road systems, which leads to the issue of defining the model describing the interactions among the intersections.

B. DDDAS-specific Problem Statement and the SBOaaS Approach

Examples, such as the traffic light for multiple intersections, say, in a city downtown, pose significant challenges due to the compute-intensive nature of the solution approach. Moreover, the dynamic nature of traffic patterns (e.g., morning and evening rush hour versus afternoon and night hours) will require periodically recomputing the optimal parameters, which further complicates the problem and its demands on resources.

Two fundamental problems exist in this realm. First, it is likely that the DDDAS feedback loop may have access to only black box models of the dynamic systems, yet will require that the DDDAS infrastructure obtain optimal parameters to be used in the DDDAS feedback loop. Second, the significantly compute intensive nature of the solution approaches makes it infeasible to deploy such model simulations in-house. Rather, there is a need for elastic computing capabilities. Thus, the DDDAS problem we solve in this paper can be posed as: (a) How to obtain the optimal parameters, and (b) How to elastically scale the compute resources as the computational needs of the solution approach dynamically changes?

This paper solves this fundamental problem using the following duo of synergistic approaches: First, we use simulations in an optimization loop to derive the best values of system parameters for a given system state particularly when the system has too many parameters and traditional means to optimize the outcomes are intractable. The approach is called *simulation-based optimization.* To address the need for elastic resources, we exploit Cloud computing as the means to address these needs and provide a framework to realize what we call *Simulation-based Optimization-as-a-Service (SBOaaS).*

Figure 7 visually represents how SBOaaS can be used to deploy the dynamic traffic light control system with online simulation-based optimization. The control system is a closed loop, periodically receiving the real time distribution of vehicle flows – which represents the dynamic and data-driven traits of DDDAS – running multiple simulations in parallel to find the optimal thresholds, and sending the feedback to the traffic light controllers – which represents the closing of the loop in DDDAS.

C. Key Features of SBOaaS

The following represent the key features of SBOaaS.

- **A cloud based solution for parallel execution of multiple simulations.** Applying computationally expensive online simulation-based optimization is usually time consuming and often fails to address the real-time constraints of applications. Moreover, for stochastic simulation models, every simulation process can vary and yield different results. To analyze the temporal properties of a stochastic system, a large number of simulation tasks needs be executed to obtain the probability distribution of simulation results. Thus, the simulation service needs to have the ability to execute multiple simulations in parallel. In our solution, to overcome this problem, we present a cloud-based approach, which is an orchestration middleware helping people to deploy DDDAS applications to the platforms of various cloud service providers without considering platform differences. It integrates the simulation manager having the capability to spawn and execute simulations in parallel and the result aggregation component using several aggregation strategies to recycle the results from the terminated simulations. A web-based interface is also implemented, which allows a user to customize both the simulation model and the input parameters, as well as to monitor the optimization process.
- **Generic problem decomposition schemes for large scale discrete variable decision problems.** In simulation-based optimization, the results of simulations are often quite different depending on the input parameters supplied to the model. To find the optimal solution, the search space sometimes can be extremely large so that such large-scale problems are intractable to naïve brute force search. In this situation, even parallel computations do not help. In our framework, a collection of generic problem decomposition schemes based on coordinate decent methods is demonstrated, which not only provides an efficient way to parallelize the optimal decision problems with discrete variable domains, but

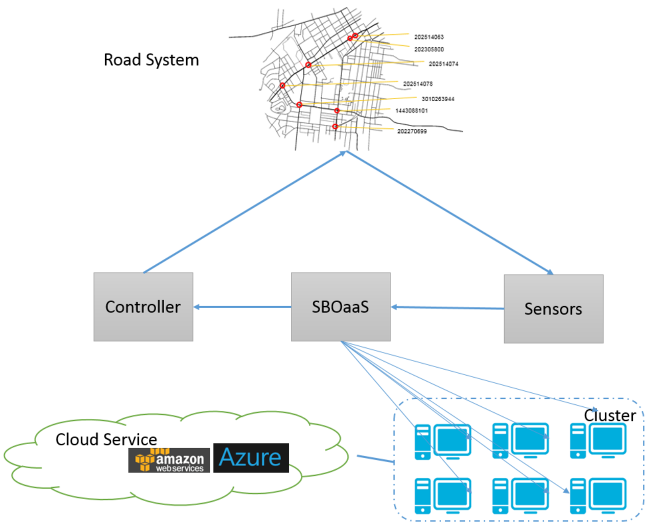


Fig. 7. SBOaaS for dynamic traffic light control system

also has the ability to execute anytime optimizations providing a flexible balance between fast response and solution quality.

- **The ability to decouple simulation based problem designs from the problem decomposition schemes.** For traditional model-based online learning and simulation approaches in DDDAS, developers usually need to face and maintain several parts of the system at different levels simultaneously. For example, there is domain-specific knowledge to setup and deploy the simulation environments, different parallelism approaches for various optimization tasks, and system management for regular maintainance. Such a method is not a good practice for a developer team that expects rapid deployment on available resources. SBOaaS leverages Linux container-based infrastructure which aims to create an abstraction layer that helps decouple simulation-based problem designs from the problem decomposition schemes. This approach allows domain experts to encapsulate the simulation environment in a container, while developers design the parallelism process according to the pre-defined interface and system administrators can simply combine both parts to run an optimization without knowing the implementation detail. Moreover, such an approach provides low runtime overhead, negligible setup and tear down costs when deploying the simulations on computing nodes, and fast data exchange among cluster hosts with incremental updates.

V. Dataflow Programming Model using Reactive Extensions

Addressing Challenge 3 requires a programming model that provides a first-class abstraction for streams; and one that is composable. Additionally, it should provide an exhaustive set of reusable coordination primitives for reception, demultiplexing, multiplexing, merging, splitting, joining two or more data streams. We argue that a dataflow programming model that provides the coordination primitives (combinators) implemented in functional programming style as opposed to an imperative programming style yields significantly improved expressiveness, composability, reusability, and scalability. A desirable solution should enable an end-to-end dataflow model that unifies the local as well as the distribution aspects.

To that end we have focused on composable event processing inspired by Reactive Programming [32] and blended it with data-centric pub/sub. Reactive programming languages provide a dedicated abstraction for time-changing values called *signals* or *behaviors*. The language runtime tracks changes to the values of signals/behaviors and propagates the change through the application by re-evaluating dependent variables automatically. Hence, the application can be visualized as a *data-flow*, wherein data and respectively changes thereof implicitly flow through the application [33], [34]. Functional Reactive Programming (FRP) [35] was originally developed in the context of pure functional language, Haskell and has since been implemented in other languages, for example, Scala.React (Scala) [16], FlapJax (Javascript) [36], Frappe (Java) [37].

Composable event processing–a modern variant (without continuous time abstraction and denotation semantics) of FRP–is an emerging new way to create scalable reactive applications [38], which are applicable in a number of domains including HD video streaming [39] and UIs. It offers a declarative approach to event processing wherein program specification amounts to "what" (i.e., declaration of intent) as opposed to "how" (looping, explicit state management, etc.). State and control flow are hidden from the programmers, which enables programs to be visualized as a data-flow. Furthermore, functional style of programming elegantly supports composability of asynchronous event streams. It tends to avoid shared mutable state at the application-level, which is instrumental for multicore scalability. Therefore, there is a compelling case to systematically blend reactive programming paradigm with data-centric pub/sub mechanisms for realizing emerging IIoT applications.

We have combined concrete instances of a publish/subscribe technology and reactive programming, to evaluate and demostrate our research ideas. The data-centric pub/sub instance we have used is OMG's DDS, more specifically the DDS implementation provided by Real Time Innovations Inc; while the reactive programming instance we have used is Microsoft's .NET Reactive Extensions (Rx.NET) [40]. Details of our approach appear in [41].

VI. Dynamic Resource Management and Dataflow Programming Models for Cloud Data Center

Addressing the challenges outlined in Section II requires a systematic and scientific approach that is reusable and easily adopted across different cloud computing platforms. To that end, this research has designed and validated a holistic set of solutions that can easily be integrated into the existing cloud computing infrastructure fabric. The key distinguishing feature of this research is that each of these solutions defines a concrete and systematic process that cloud service providers, including DoD cloud platforms, can employ for their cloud platforms. Although our solutions were designed and validated in a private data center virtualized by the Xen hypervisor and managed by OpenNebula cloud management tool, the principles behind the solutions are broadly applicable.

Many of our techniques are based on learning the model of the cloud. To that end, we have used different machine learning techniques for learning and implemented the control algorithms for managing the resources in the cloud data center.

A. Addressing Challenge 1 → iTune: Engineering the Performance of Xen Hypervisor via Autonomous and Dynamic Scheduler Reconfiguration

To address challenge 1, we have developed iTune, which is a middleware that optimizes the Xen hypervisor's scheduler configuration parameters autonomously through a three phase design workflow comprising: (1) Discoverer, which monitors and saves the resource usage history of the host machines and groups set of related host machine workload, (2) Optimizer, where optimum Xen scheduler configuration parameters for each workload cluster is explored by employing a simulated annealing machine learning algorithm, and (3) Observer, where iTune monitors the resource usage of host machines online, classifies them into one of thecategories found in the Discoverer phase, and loads the optimum scheduler parameters determined in the Optimizer phase.

A resource scheduler, such as the Xen credit scheduler, is a critical component of systems software that manages the resources on cloud platforms. Its design and how it manages the resources dictate the performance delivered to applications hosted in the VMs in individual Xen domains. The scheduler's resource management behavior depends on how it is configured in terms of its parameters, which is the responsibility of the cloud operator managing the platform. The operator is responsible for selecting the right values for the parameters to suit the expected loads on the cloud platform.

This is a hard problem to address because the number of configuration parameters and their available ranges give rise to a total of roughly $65535 \times 1200 \times 499900 \times 1000 = 3.9 \times 10^{16}$ different configuration settings for a 12 CPU host machine. Relying on the default values of each parameter may not always work well for every application type and workload on a host machine. While a rate limit value less than 1,000 microseconds could work well for latency-sensitive applications, it might not work well for CPU-intensive applications. Thus, application developers interested in deploying their applications in the virtualized cloud platforms must determine the best configuration settings for their applications. Moreover, they need to determine how these parameters must be changed at runtime as the system dynamics change due to workload and resource availability changes. Addressing these challenges in an automated way so that the system administrator is relieved of these responsibilities is the focus of our research.

Figure 8 depicts the three distinct phases of iTune which are encoded in the following components: (1) Discoverer, (2) Optimizer, and (3) Observer. Figure 9 depicts the system architecture. The work is described in more details in [42].

B. Addressing Challenge 2 → iOverbook: Intelligent Resource-Overbooking to Support Soft Real-time Applications in the Cloud

To address Challenge 2, we have developed iOverbook, which is an overbooking strategy that uses a machine learning approach to make systematic and online determination of overbooking ratios such that the quality of service needs of soft real-time systems can be met while still benefiting from overbooking. Specifically, iOverbook utilizes historic data of tasks and host machines in the cloud to extract their resource usage patterns and predict future resource usage along with the expected mean performance of host machines. To evaluate our approach, we have used a large usage trace made available by Google of one of its production data centers.

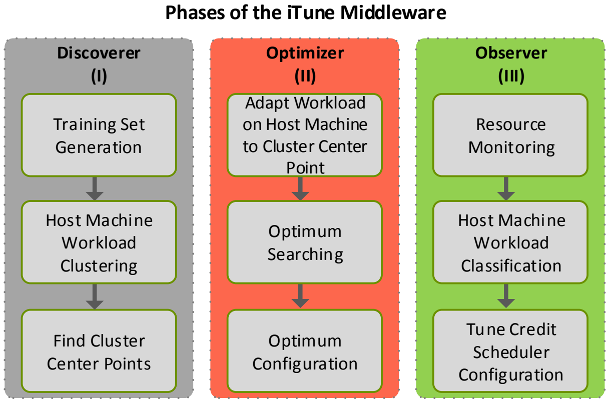

Phases of the iTune Middleware
Discoverer (I)
Training Set Generation
Host Machine Workload Clustering
Find Cluster Center Points
Optimizer (II)
Adapt Workload on Host Machine to Cluster Center Point
Optimum Searching
Optimum Configuration
Observer (III)
Resource Monitoring
Host Machine Workload Classification
Tune Credit Scheduler Configuration

Fig. 8. Three distinct phases of iTune

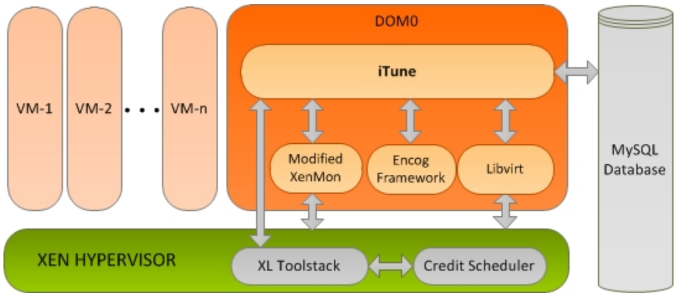


Fig. 9. iTune Architecture

Figure 10 depicts the architecture of iOverbook, which is our intelligent, machine learning-based approach for online determination of effective overbooking ratios for the machines of a data center. Specifically, we focus on the CPU and memory overbooking ratios for each individual host machine within a specified future time interval. Since the online computation of effective overbooking ratios must assure the performance of soft real-time applications, we require an understanding of how the resources are currently utilized and the properties of existing applications so that we can predict the resource usage for a future specified time interval. Once we know this information, we can determine how much overbooking is feasible and whether it is acceptable for soft real-time applications or not.

These responsibilities motivated a three stage design for iOverbook, which comprises: (1) a resource usage predictor, (2) an overbooking ratio prediction engine, and (3) a performance assessor. The resource usage predictor and performance assessor components retrieve historic data from a training set repository to train their internal neural networks. iOverbook utilizes mean CPU and memory request, mean CPU and memory usage, mean performance, mean VM count, mean CPU and memory capacity, and CPU and memory overbooking ratios as input parameters. iOverbook is described in more details in [43].

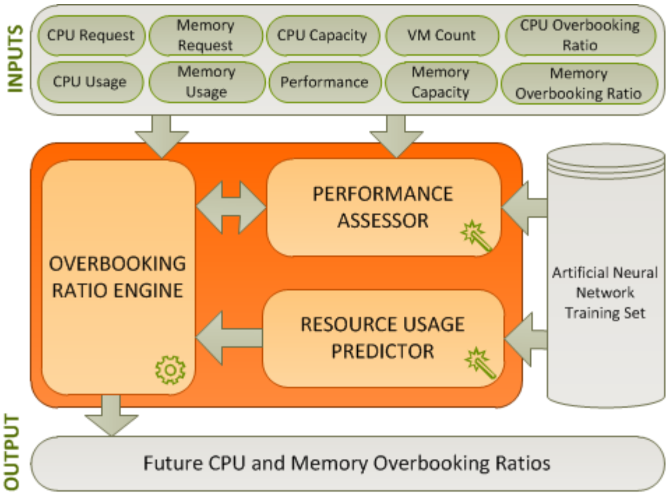


Fig. 10. iOverbook System Architecture

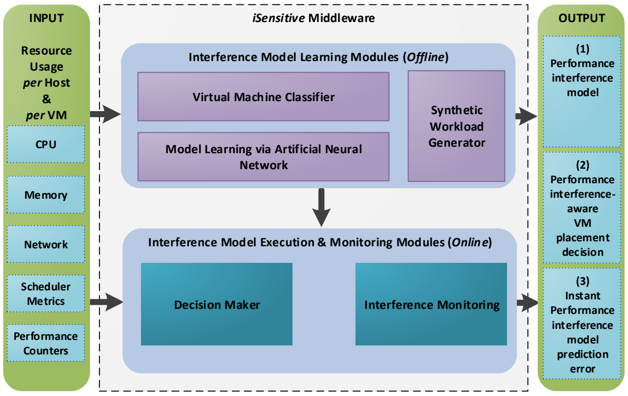

INPUT
Resource Usage *per* Host & *per* VM
CPU
Memory
Network
Scheduler Metrics
Performance Counters
iSensitive Middleware
Interference Model Learning Modules (*Offline*)
Virtual Machine Classifier
Model Learning via Artificial Neural Network
Synthetic Workload Generator
Interference Model Execution & Monitoring Modules (*Online*)
Decision Maker
Interference Monitoring
OUTPUT
(1) Performance interference model
(2) Performance interference-aware VM placement decision
(3) Instant Performance interference model prediction error

Fig. 11. Conceptual Design of iSensitive Illustrating Input, Output, and System of Interest

C. Addressing Challenge 3 → iSensitive: An Intelligent Performance Interference-aware Virtual Machine Migration Approach

Resource contention and hence performance interference is unavoidable in virtualized environments due to the nature of resource sharing. We have validated this assumption empirically where we analyzed how performance interference stems from resource overbooking and how contention impacts the application performance running in the VMs managed by KVM hypervisor.

To address these issues described in Challenge 3, we have developed iSensitive, which is a machine learning-based middleware providing an online placement solution where the system is trained using events and lifecycle of a publicly available trace of a large data center owned by Google. Our approach first classifies the VMs based on their historic mean CPU, memory usage, and network usage features. Subsequently, it learns the best patterns of collocating the classified VMs by employing machine learning techniques. These extracted patterns document the lowest performance interference level on the specified host machines making them amenable to hosting applications while still allowing resource overbooking.

Figure 11 shows the algorithmic design and building blocks of our framework called iSensitive that adopts the solution approach described above. As shown, iSensitive comprises two distinct modules: (1) Interference Model Learning Module (*offline*), and (2) Interference Model Execution and Monitoring Module (*online*). The Interference Model Learning Module in turn comprises three main components: (1) Virtual Machine Classifier, (2) Model Learning via Artificial Neural Network, and (3) Synthetic Workload Generator. The Interference Model Execution and Monitoring Module consists of two primary components: (1) Decision Maker, and (2) Interference Monitoring.

Since resource utilization is a key indicator of performance interference, iSensitive utilizes different resource usage metrics, such as CPU usage, memory usage, network I/O usage, internal scheduler metrics, hardware and kernel-level performance counters for VMs and physical host as input to the system. These metrics are retrieved with the help of (1) *perf,* performance analyzing tool in Linux, (2) *mpstat,* Linux command for processor related statistics, and (3) *libvirt,* toolkit to interact with the underlying virtualization system. The *virtual machine classifier* clusters VMs into similar sets of objects by employing the k-means algorithm and the silhouette method. These classes of VMs are then used by the *artificial neural network* to extract the "best collocated VM patterns", which are those that lead to minimal performance interference on the host machines. In other words, a performance interference model of a host machine is generated.

After the neural network is trained, the *decision maker* is employed to find the aptly suited host machine having the minimal performance interference by utilizing the trained model. *Interference monitoring* is responsible to compare the actual performance interference value and its predicted value. If the difference is greater than a threshold value, then that collocation pattern is saved for future model refinements.

Note that for our work, we have assumed that the physical host machines in the cloud data center are homogeneous and therefore a model generated for one host machine is applicable to all other physical hosts. If a data center comprises heterogeneous machine types, then performance interference models for each different host machine type must be created.

D. Addressing Challenge 4 → iPlace: An Intelligent and Tunable Power- and Performance-Aware Virtual Machine Placement Technique for Cloud-based Real-time Applications

To address Challenge 4, we have developed iPlace, which is a middleware providing an intelligent and tunable power- and performance-aware VM placement capability. The placement strategy is based on a two-level artificial neural network, which predicts (1) CPU usage at the first level, and (2) power consumption and performance of a host machine at the second level that uses the predicted CPU usage. The placement decision (i.e., aptly suited host machine for the VM being deployed) is determined by making the appropriate trade-offs between predicted power and performance values of a host machine.

Figure 12 depicts the strategy of iPlace, which is our intelligent power- and performance-aware virtual machine placement algorithm. The goal of iPlace is to find an aptly suited host machine by carefully considering the energy efficiency of the data center and performance requirements of soft-real time applications running on host machines. iPlace takes power changes and performance effects to the applications running on VMs for its placement decision. A tunable parameter named *performance preference level* is provided to iPlace in advance to set the performance requirement.

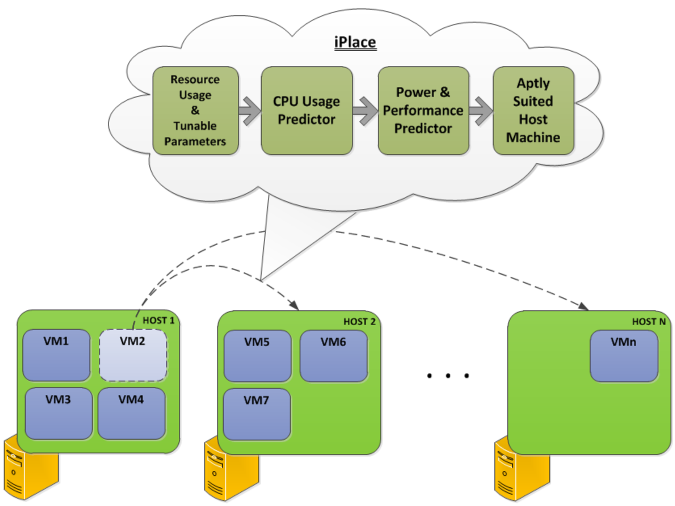

iPlace
Resource Usage & Tunable Parameters
CPU Usage Predictor
Power & Performance Predictor
Aptly Suited Host Machine
HOST 1
VM1
VM2
VM3
VM4
HOST 2
VM5
VM6
VM7
. . .
HOST N
VMn

Fig. 12. Illustration of iPlace's Virtual Machine Placement Strategy

To find the aptly suited host machine, a two-level artificial neural network (ANN) is employed by our VM placement middleware, which are at the core of our system design and serve as the predictor mechanism. To train the ANNs, iPlace employs the Levenberg-Marquardt back-propagation algorithm [44]. At the first level, the mean CPU usage of a host machine after a VM were to be migrated to it is predicted by running the CPU usage predictor ANN. Subsequently, this predicted CPU usage value is utilized by the second level ANN. At the second level, power consumption and mean performance of the host machine is predicted by the power and performance predictor ANN. At runtime, the middleware will consult the prediction engine and if the predicted values are acceptable, the middleware will take the decision of placing the VM on a given host.

To understand how these ANNs are used to make runtime decisions, consider the case when one of the consolidation algorithms, high availability solutions, or scheduling mechanisms would like to migrate a VM from one host machine to another one. iPlace finds the aptly suited host machine by predicting the power consumption and performance values for each host machines in the cluster as though the VM was migrated on to it. As illustrated in Figure 12, iPlace employs both CPU usage predictor and power and performance predictor sequentially by feeding their required input values.

In our current design, iPlace targets only compute-intensive applications, therefore $1/(CPUtime)$ metric was utilized in this work as the performance indicator of an application. The higher the performance value, the better the performance. Additionally, we assume that CSPs overbook their underlying cloud infrastructure to save energy costs. Details of the ANNs are described below.

E. Addressing Challenge 5 → Stochastic Model Checking using Lightweight Virtualization

A cloud platform is an attractive choice to address Challenge 5 because it can elastically and on-demand execute the multiple different simulation trajectories of the simulation models in parallel, and perform aggregation such as stochastic model checking (SMC) to obtain results within a desired confidence interval. The challenge stems from provisioning these simulation trajectories in the cloud in real-time so that the response times perceived by the user are acceptable. To that end we have architected the SIMaaS cloud-based simulation-as-a-service (SIMaaS) and its associated middleware as shown in Figure 13.

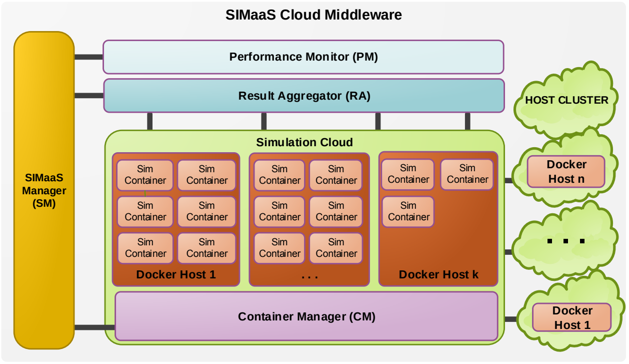


Fig. 13. System Architecture

The SIMaaS cloud middleware leverages Linux container [45]-based infrastructure, which has low runtime overhead, higher level of resource sharing, and very low setup and tear down costs. It provides a resource management algorithm, that reduces the cost to the service provider and enhances the parallelization of the simulation jobs by fanning out more instances until the deadline is met while simultaneously auto-tuning itself based on the feedback. The SIMaaS middleware intelligently generates different configurations for experimentation, and intelligently schedules the simulations on the Linux container-based cloud to minimize cost while enforcing the deadlines.

Using two case studies, we show the viability of a Linux container-based SIMaaS solution, and illustrate the performance gains of a Linux container-based approach over hypervisor-based traditional virtualization techniques used in the cloud. Details on the SIMaaS approach appears in [46].

VII. New Research Directions: DDDAS for Cloud/Fog/Edge Computing

A. Emerging Trends

The elastic properties and cost benefits of the cloud has made it an attractive hosting platform for a variety of soft real-time cyber physical systems (CPS)/Internet of Things (IoT) applications, such as cognitive assistance, patient health monitoring and industrial automation. The stringent quality of service (QoS) considerations of these applications mandate both predictable performance from the cloud and lower end-to-end network latencies between the end user and the cloud. To date, security and performance assurance continues to be a hard problem to resolve in cloud platforms due to their virtualized and multi-tenant nature [47]. Although recent advances in fog and edge computing have enabled cloud resources to move closer to the CPS/IoT devices thereby mitigating the network latency concerns to some extent [48], there is still a general lack of mechanisms that can dynamically manage resources across the cloud-edge spectrum. This is a hard problem to resolve due to the highly dynamic behaviors of the edge and cloud. Consequently, any pre-defined and fixed set resource management policies will be rendered useless for hosting CPS/IoT applications in the cloud.

The dynamic data driven application systems (DDDAS) paradigm [8] addresses precisely these challenges. DDDAS prescribes an approach where applications are instrumented adaptively so that their models can be learned and enhanced continuously, and in turn these models can be analyzed and used in a feedback loop to steer the applications along their intended trajectories. Previous work have focused on a specific application or applied DDDAS for resilience and security [49]. We propose to apply the DDDAS principle to the pool of resources spanning the cloud-edge spectrum to enable and enforce dynamic resource management decisions that deliver the required QoS properties of cloud-hosted applications. To that end we propose Dynamic Data Driven Cloud and Edge Systems (D^3CES), which uses performance data collected from adaptively instrumenting the cloud and edge resources to learn and enhance models of the distributed resource pool, and in turn using these models in a feedback loop to make effective resource management decisions to host CPS applications and deliver their QoS properties.

B. Key Research Challenges and Solution Needs

Our research calls for an effective use of resources across the cloud data centers (CDCs) and the micro data centers (MDCs) that reside at the edge. The following lists a non-exhaustive set of challenges along three dimensions that we are addressing in this proposed research.

1) Application-imposed Challenges:

1) **Workload variations:** The workload generated by CPS/IoT applications may illustrate both transient and sustained variability which needs to be predicted and addressed.
2) **Stochastic execution semantics:** For some CPS/IoT applications, their uncertain and dynamic nature may require several instances of the same tasks to be executed to reach specified confidence levels. Each execution may take different execution times but impose certain QoS needs.
3) **Application structure:** Increasingly, cloud-based applications are realized as a collection of communicating microservices, which can be deployed independently across the spectrum of resources. This gives rise to interesting challenges in whether part or entire service must be migrated closer to the edge.
4) **Reconciling application state:** When a cloud-hosted application is migrated to a MDC, often not all of its state may be transferred to the MDC and hence may have to be reconciled periodically with the state maintained at the CDC, which gives rise to interesting consistency versus availability tradeoffs.
5) **High degree of user mobility:** CPS/IoT systems, such as autonomous transport vehicles, unmanned aerial vehicles, and mobile devices, operate in a highly uncertain environments with dynamic movement profiles. Thus, a designated edge resource cannot serve such users for long durations of times.

2) Cloud Provider-related Challenges:

1) **Effective utilization of edge resources:** Although exploiting edge resources is an intuitive solution to addressing the network latency issues, the MDCs will also face the same challenges as a CDC, which stem from virtualization and multi-tenancy resulting in application performance interference [50].
2) **Workload consolidation and migration across MDCs:** Since the edge may comprise multiple MDCs, there is a need for effective and dynamic server workload consolidation across MDCs and CDCs.
3) **Distributed user base:** Collaborative applications such as online games may often involve a distributed set of users. Consequently, determining the MDC to migrate the application to and whether to migrate it to multiple MDCs remains an open question.
4) **Shared micro data centers:** In the simplest case, an edge-based MDC may be considered to be owned by the same provider that owns a CDC. In general, however, a MDC could be shared across different CDC providers. Assuring security and isolation guarantees in these scenarios is an open question.
5) **Energy savings and revenue generation:** In making use of the spectrum of resources across the cloud and the edge, a cloud provider will be concerned about maximizing revenues and conserving energy while ensuring that application SLOs are met.

3) Measurement-related Challenge:

1) **Collecting metrics under hardware heterogeneity:** The plethora of deployed hardware configurations with different architectures and versions makes it hard to

collect various performance metrics. Modern architectures are making it easier to collect more finer grained performance metrics, however, much more research is needed in identifying effective approaches to control the hardware and derive the best performance out of them.

2) **Lack of benchmarks:** There is a general lack of open source and effective benchmarking suites that researchers can use to conduct studies and build models of the cloud-edge spectrum of resources that then can be used in resource management.

C. Ongoing Work

In ongoing work [51] we are focusing on addressing the key factors that affect the round trip latencies, specifically the roundtrip delay between the nearest access point of the IoT end user and the cloud, and the time it takes to serve the client request in the cloud. Thus, any improvement in round trip latencies revolves around reducing the network delays and the server processing time. To that end we are exploiting advances in fog/edge computing, such as cloudlets or micro data centers (MDCs) [52].

A fundamental system property that is often overlooked in related research that use fog resources is performance interference, which is caused by co-located applications in virtualized data centers [53], [54]. Performance interference being an inherent property of any virtualized system, it manifests itself in MDCs also and therefore must be factored in any approach that is performance-aware. Thus, our ongoing work is focusing on a "just-in-time and performance-aware" service migration approach for moving cloud-based services for the assistive applications to a MDC. A number of challenges including the heterogeneity in the hardware, and difficulty in measuring performance interferences and other system and network performance metrics must be overcome. We are addressing these challenges in the context of providing a ubiquitous deployment approach that spans the cloud-edge spectrum to support the safety critical IoT applications.

Our future directions in this space include considering variable workloads and user mobility, which will manifest when multiple different assistive applications co-exist and where the end users are mobile. This requires building a profile of users and IoT applications to forecast the load and expected network latency and bandwidth, and employ efficient resource management algorithms. It will also require on-demand workload consolidation and service migration which builds on our existing work that moves the service from the cloud to the fog without any reconfiguration thereafter.

We are incorporating serverless computing and micro-services as part of our research because we believe that IoT-based applications are likely to be developed as a composition of microservices that execute on the heterogeneous IoT resources. These micro services may have dependencies on each other and their states have to be managed while distributing and migrating them across the central cloud and edge resources for optimal performance. The micro services can be packaged as self-contained deployable units using Linux containers such as Docker or Unikernels to address the heterogeneity and orchestration issues. As part of the research, we are developing algorithms to perform global optimization to answer *if* and *when* should we migrate the services, identify the nearest edge cloud and how to do this efficiently.

To validate our claims, we are setting up a large IoT testbed with a variety of edge resources (e.g., Raspberry PI-3, BeagleBone Black, Intel Edisons, DecaWave sensors, Minnowboards that can run Docker containers, specialized devices such as SmartEyeGlasses) and cloud resources involving latest hardware advances, such as cache allocation technologies.

VIII. Concluding Remarks

This report presented progress made by the Vanderbilt University's DDDAS project called AMASS during the three years of the project.

A. Summary of Research Contributions

Our research contributions can be summarized as follows:

Contribution 1: Gaussian Process Modeling for Workload Characterization: To address the Challenge 1 listed in Section II-A requires a runtime performance model of the system so that runtime decisions on VM placement and migration can be made by a controller that incorporates the model. In this work, we propose a model-based data-driven approach that abstracts the runtime behavior and characteristics of different collocated workloads, which is known to impact the performance interference level. Recent research efforts have applied Big Data analytics methodologies to analyze and model the cloud infrastructure so that businesses can utilize autonomous machine-based decision making solutions. To that end, our approach uses a machine learning algorithm to learn the online performance model of the collocated workloads based on measured data to provide real time predictive analysis of performance interference. Moreover, our approach relearns or updates the predictive model online in order to overcome run-time VM workload changes.

Our approach consists of twofold. First, we select data features and build the model. To do so, we need to analyze the data and the correlation between them, then select the correlated data features to construct the model with. Lastly, we use a Gaussian Process (GP) model as a data-driven machine-learning approach to learn and train the model from the data. Second, we use the GP to construct the predictive distribution of the system performance metrics where we use this distribution for developing an autonomous machine-based VM placement/migration decision to minimize the system cost represented in the interference level.

Contribution 2: Simulation-based Optimization-as-a-Service: In addressing the Challenge 2 listed in Section II-B, we exploit cloud computing, which provides an economical solution for individuals and organizations with limited resources to execute compute-intensive tasks, which has become a highly demanded utility due to the advantages of potentially unlimited computing power available on-demand, affordable cost of services without incurring any capital and operation expenditures, elasticity of resources, and its ability to autoscale on demand. Thus, cloud-based simulation services have opened up new avenues to address the challenges stemming from the simulation based optimizations noted above.

To address the known challenges with simulation-based optimizations while exploiting emerging computing paradigms, we have developed a cloud-based framework that provides a “simulation-based optimization as a service (SBOaaS),” in which real-time considerations are explicitly accounted for making optimal use of limited but parallel computational resources in order to obtain the best answer in the given time constraints. Specifically, in this paper we present a generic optimization process for deploying simulation-based optimization on a cloud architecture. Our framework consists of (a) the implementation of SBOaaS, which describes for a given optimization problem, how to decompose the input problem into a group of parallel simulations and efficiently use the existing computing power; and (b) an anytime parallel simulation-based optimization approach, which admits significant flexibility in both time and computational resource constraints to obtain the best (but possibly suboptimal) solutions given the available resources and time constraints on decisions.

Contribution 3: Scaling Dataflow Programming Models: Reactive programming is increasingly becoming important in the context of real-time stream processing for big data analytics that employ dataflow parallel programming models. Reactive programming supports four key traits: event-driven, scalable, resilient and responsive. While reactive programming is able to support these properties, most of the generated data must be disseminated from a large variety of sources (*i.e.*, publishers) to numerous interested entities, called subscribers while maintaining anonymity between them. These properties are provided by pub/sub solutions, such as the OMG DDS, which is particularly suited towards real-time applications. Bringing these two technologies together helps solve both the scale-out problem (*i.e.*, by using DDS) and scale-up using available multiple cores on a single machine (*i.e.*, using reactive programming).

To that end and address Challenge 3 from Section II-C, our work integrated the Rx .NET reactive programming framework with OMG DDS, which resulted in the RxDDS.NET library. To understand the advantages gained by this effort, we have used the DEBS 2013 grand challenge problem to compare a solution that uses RxDDS with a plain, imperative solution we developed using DDS and C++11, and made qualitative comparisons between these two efforts.

Contribution 4: Dynamic Resource Management: We have addressed a range of dynamic resource management problems collected under Challenge 4 in Section II-D as part of our research. This research was motivated by the need for innovative solutions to address dynamic resource management and energy conservation challenges in cloud data centers, specifically focusing on the virtualization, cloud management, application and service delivery layers. To that end we developed a set of novel solutions each of which addresses a specific set of challenges. Each of these solutions provides a systematic and scientific approach that a cloud service provider can implement in their data centers to address energy consumption and resource utilization challenges. The individual solutions comprised:

1) **Autonomous and Dynamic Reconfiguration of Hypervisor Scheduler.** The challenges in the area of autonomous and dynamic scheduler reconfiguration are addressed by *Engineering the Performance of Xen Hypervisor via Autonomous and Dynamic Scheduler Reconfiguration* middleware, called iTune. iTune automatically reloads the optimum configuration based on the changing workload on the host machine. iTune comprises three phases named Discoverer, Optimizer, and Observer and employs machine learning algorithms. iTune provides options to mark the VMs into one of the four latency sensitivity categories (i.e. LS-1, LS-2, LS-3, and NLS). This allows iTune to assure performance requirements associated with these latency sensitivity levels.
   Although iTune has currently been demonstrated in the context of the Xen credit scheduler, testing the approach and comparing the results for other systems software are left as a future work. Additionally, the number of regions of operation (i.e., clusters) for training set is based on a specific workload we generated. Hence, we suggest that CSPs first apply iTune to their historic workloads.
2) **Resource-Overbooking to Support Soft Real-time Applications.** The challenges in the area of dynamic resource-overbooking are addressed by *Intelligent Resource-Overbooking to Support Soft Real-time Applications in the Cloud*, called iOverbook. iOverbook determines the CPU and memory overbooking ratios for each host machine in the cloud by predicting their future resource usage demands, and considering the QoS requirements of soft real-time applications. The benefits and efficacy of iOverbook were evaluated in the context of resource utilization and energy efficiency in the data centers by utilizing Google’s cluster trace log data. Our future work for iOverbook will investigate effective filtering of outliers and using confidence intervals.
3) **Performance Interference Effects on Application Performance.** The challenges in the area of performance interference effects on application performance are addressed by *An Intelligent Performance Interference-aware Virtual Machine Migration Approach*, called iSensitive. The proposed research investigated the performance interference effects on application performance and creating a model to make intelligent virtual machine placement.
   Presently, iSensitive does not consider disk-intensive applications. Disk-intensive applications need to be considered for the systems utilizing local disk. Additionally, analyzing energy efficiency and performance interference properties should also be considered.
4) **Power- and Performance-Aware Virtual Machine Placement.** The challenges in the area of power- and

performance-aware virtual machine placement are addressed by *An Intelligent and Tunable Power- and Performance-Aware Virtual Machine Placement Technique for Cloud-based Real-time Applications*, called iPlace. iPlace employs two-level artificial neural networks to predict a host machine's CPU usage at the first level and power consumption and performance of the host machine at the second level. In its current form, iPlace targets only the compute-intensive applications due to the metrics utilized. Supporting variety of application types in the cloud environment should be considered.

5) **Container-based Deployment for Stochastic System Models:** Our solution described the design and empirical validation of a cloud middleware solution to support the notion of simulation-as-a-service. Our solution is applicable to those systems whose models are stochastic and require a potentially large number of simulation runs to arrive at outcomes that are within statistically relevant confidence intervals, or systems whose models result in different outcomes for different parameters. Our solutions uses lightweigth virtualization in the form of Docker containers and provides resource management solutions in that context.

B. Research Outcomes

Our research contributions can be summarized as follows:

1) **Dissertation:** One PhD dissertation resulted from this effort. Faruk Caglar defended his PhD dissertation in April 2015. One proposal defense (Hamzah Abdul Aziz, March 2017) was successful. Another two proposal defenses are scheduled (Shashank Shekhar and Shweta Khare, Summer 2017).
2) **Workshop/Panel:** PI Aniruddha Gokhale participated in the DDDAS/Infosymbiotics panel at Supercomputing 2014. PI Gokhale was also a co-organizer of the DDDAS/Infosymbiotics workshop held at the HiPC conference in Dec 2015.
3) **Publications:** Multiple journal, conference and workshop publications resulted from this work, which have either appeared or are currently in submission as listed here [55], [56], [57], [58], [43], [59], [41], [42], [46], [54], [30], [51].
4) **Software and Algorithms:** Most of our work to date is available for download. For instance, our ongoing work on edge/fog/cloud is available at https://github.com/shekharshank/indices. Our ongoing work is focusing on packaging all of the deliverables as part of a single framework and make it available in a common githib repository under https://github.com/orgs/doc-vu.

C. Ongoing and Follow-on Research

1) **Developing new ideas for Infosymbiotics at the Edge:** To continue our work in the DDDAS area, we are exploring new research dimensions for Infosymbiotics at the edge where processing must be carried out in the context of the mobile, resource-constrained devices of different modalities.
2) **Leverage DURIP Award for Infosymbiotics at the Edge:** Our recent AFOSR funded DURIP award is enabling us to set up a testbed to test our ongoing and future ideas, particularly on the Cloud-Edge continuum.
3) **Outreach:** PI Gokhale has teamed up with other PIs Dr. Vaidy Sunderam (Emory), Dr. Sandu (Virginia Tech) and Dr. Hariri (Arizona) to guest edit a DDDAS special issue of Springer Cluster Computing. The guest issue will be published mid 2017.

Acknowledgments

A major part of this work was supported by AFOSR DDDAS Grant FA9550-13-1-0227. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of AFOSR. We are thankful to the Program Director of DDDAS Program, Dr. Frederica Darema, other DDDAS PIs, and anonymous reviewers of our publications for their helpful feedback that helped improve the quality of our research.

We also thank our collaborators Dr. Yevgeniy Vorobeychik and Dr. Abhishek Dubey, and students Yi Li, Anirban Bhattacharjee, Yogesh Barve and Violetta Vyleghzhanina for their help on collaborative efforts.

References

[1] N. Tomizawa, "On some techniques useful for solution of transportation network problems," *Networks*, vol. 1, no. 2, pp. 173–194, 1971.

[2] H. Chourabi, T. Nam, S. Walker, J. R. Gil-Garcia, S. Mellouli, K. Nahon, T. A. Pardo, and H. J. Scholl, "Understanding smart cities: An integrative framework," in *System Science (HICSS), 2012 45th Hawaii International Conference on*. IEEE, 2012, pp. 2289–2297.

[3] I. C. Inc., "What happens in an internet minute?" http://www.intel.com/content/www/us/en/communications/internet-minute-infographic.html, 2014.

[4] C. O'Luanaigh, "Cern data center passes 100 petabytes," http://home.web.cern.ch/about/updates/2013/02/cern-data-centre-passes-100-petabytes, 2013.

[5] US Environmental Protection Agency, "Report to Congress on Server and Data Center Energy Efficiency: Public Law 109-431," 2007.

[6] J. Koomey, "Growth in data center electricity use 2005 to 2010," *A report by Analytical Press, completed at the request of The New York Times*, 2011.

[7] K. Brandt, "Better buildings challenge expands to take on data centers," http://www.whitehouse.gov/blog/2014/09/30/better-buildings-challenge-expands-take-data-centers, 2014.

[8] F. Darema, "Dynamic Data Driven Applications Systems: A New Paradigm for Application Simulations and Measurements," *Computational Science-ICCS 2004*, pp. 662–669, 2004.

[9] M. C. Fu and J.-Q. Hu, "Sensitivity analysis for monte carlo simulation of option pricing," *Probability in the Engineering and Informational Sciences*, vol. 9, no. 03, pp. 417–446, 1995.

[10] G. Gurkan, A. Y. Ozge, and T. Robinson, "Sample-path optimization in simulation," in *Simulation Conference Proceedings, 1994. Winter*. IEEE, 1994, pp. 247–254.

[11] P. Kim and Y. Ding, "Optimal engineering system design guided by data-mining methods," *Technometrics*, vol. 47, no. 3, pp. 336–348, 2005.

[12] E. L. Plambeck, B.-R. Fu, S. M. Robinson, and R. Suri, "Throughput optimization in tandem production lines via nonsmooth programming," in *Proceedings of the 1993 Summer Computer Simulation Conference*, 1993, pp. 70–75.

[13] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," *ACM Comput. Surv.* [Online]. Available: http://doi.acm.org/10.1145/857076.857078

[14] OMG, “The Data Distribution Service specification, v1.2,” http://www.omg.org/spec/DDS/1.2, 2007.

[15] “The Reactive Manifesto,” http://www.reactivemanifesto.org, 2013.

[16] I. Maier and M. Odersky, “Deprecating the Observer Pattern with Scala.react,” Tech. Rep., 2012.

[17] L. Atzori, A. Iera, and G. Morabito, “The Internet of Things: A Survey,” *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[18] A. Alamri, W. S. Ansari, M. M. Hassan, M. S. Hossain, A. Alelaiwi, and M. A. Hossain, “A Survey on Sensor-cloud: Architecture, Applications, and Approaches,” *International Journal of Distributed Sensor Networks*, vol. 2013, 2013.

[19] V. Mauch, M. Kunze, and M. Hillenbrand, “High Performance Cloud Computing,” *Future Generation Computer Systems*, vol. 29, no. 6, pp. 1408–1416, 2013.

[20] M. García-Valls, T. Cucinotta, and C. Lu, “Challenges in Real-Time Virtualization and Predictable Cloud Computing,” *Journal of Systems Architecture*, 2014.

[21] R. M. Fujimoto, A. W. Malik, and A. Park, “Parallel and Distributed Simulation in the Cloud,” *SCS M&S Magazine*, vol. 3, pp. 1–10, 2010.

[22] R. Ledyayev and H. Richter, “High Performance Computing in a Cloud Using OpenStack,” in *CLOUD COMPUTING 2014, The Fifth International Conference on Cloud Computing, GRIDs, and Virtualization*, 2014, pp. 108–113.

[23] Z. Li, X. Li, T. Duong, W. Cai, and S. J. Turner, “Accelerating Optimistic HLA-based Simulations in Virtual Execution Environments,” in *Proceedings of the 2013 ACM SIGSIM conference on Principles of advanced discrete simulation*. ACM, 2013, pp. 211–220.

[24] X. Liu, X. Qiu, B. Chen, and K. Huang, “Cloud-based simulation: the state-of-the-art computer simulation paradigm,” in *Proceedings of the 2012 ACM/IEEE/SCS 26th Workshop on Principles of Advanced and Distributed Simulation*. IEEE Computer Society, 2012, pp. 71–74.

[25] R. M. Fujimoto, “Parallel Discrete Event Simulation,” *Communications of the ACM*, vol. 33, no. 10, pp. 30–53, 1990.

[26] K. Vanmechelen, S. De Munck, and J. Broeckhove, “Conservative Distributed Discrete Event Simulation on Amazon EC2,” in *Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012)*. IEEE Computer Society, 2012, pp. 853–860.

[27] F. Tao, L. Zhang, V. Venkatesh, Y. Luo, and Y. Cheng, “Cloud manufacturing: a computing and service-oriented manufacturing model,” *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 225, no. 10, pp. 1969–1976, 2011.

[28] H. A. Aziz, S. Shekhar, X. Koutsoukos, and A. Gokhale, “Online Performance Model Learning for Dynamic Resource Management in Cloud Computing Infrastructure,” *In submission*, 2017.

[29] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[30] Y. Li, S. Shekhar, Y. Vorobeychik, X. Koutsoukos, and A. Gokhale, “Simulation-based Optimization as a Service for Dynamic Data-driven Applications Systems,” in *To Appear in the 1st Conference on Dynamic Data Driven Applications and Systems (DDDAS)*. Hartford, CT, USA: Springer, Aug. 2016.

[31] J. Lima Fleck and C. G. Cassandras, “Infinitesimal perturbation analysis for quasi-dynamic traffic light controllers,” in *Discrete Event Systems*, vol. 12, 2014, pp. 235–240.

[32] E. Bainomugisha, A. L. Carreton, T. v. Cutsem, S. Mostinckx, and W. d. Meuter, “A survey on reactive programming,” *ACM Comput. Surv.*, vol. 45, no. 4, pp. 52:1–52:34, Aug. 2013. [Online]. Available: http://doi.acm.org/10.1145/2501654.2501666

[33] G. Salvaneschi, P. Eugster, and M. Mezini, “Programming with implicit flows,” *IEEE Software*, vol. 31, no. 5, pp. 52–59, 2014. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/MS.2014.101

[34] G. H. Cooper and S. Krishnamurthi, “Embedding Dynamic Dataflow in a Call-by-value Language,” in *Programming Languages and Systems*. Springer, 2006, pp. 294–308.

[35] C. Elliott and P. Hudak, “Functional Reactive Animation,” in *ACM SIGPLAN Notices*, vol. 32, no. 8. ACM, 1997, pp. 263–273.

[36] L. A. Meyerovich, A. Guha, J. Baskin, G. H. Cooper, M. Greenberg, A. Bromfield, and S. Krishnamurthi, “Flapjax: A Programming Language for Ajax Applications,” in *ACM SIGPLAN Notices*, vol. 44, no. 10. ACM, 2009, pp. 1–20.

[37] A. Courtney, “Frapp&#233;: Functional reactive programming in java,” in *Proceedings of the Third International Symposium on Practical Aspects of Declarative Languages*, ser. PADL ’01. London, UK, UK: Springer-Verlag, 2001, pp. 29–44. [Online]. Available: http://dl.acm.org/citation.cfm?id=645771.667929

[38] D. Synodinos, “Reactive Programming as an Emerging Trend,” http://www.infoq.com/news/2013/08/reactive-programming-emerging, 2013.

[39] “Reactive Programming at Netflix,” http://techblog.netflix.com/2013/01/reactive-programming-at-netflix.html, 2013.

[40] “The Reactive Extensions (Rx),” http://msdn.microsoft.com/en-us/data/gg577609.aspx.

[41] S. Khare, K. An, S. Tambe, A. Gokhale, and A. Meena, “Industry Paper: Reactive Stream Processing for Data-centric Publish/Subscribe,” in *The 9th ACM International Conference on Distributed Event-Based Systems (DEBS’ 15)*. Oslo, Norway: ACM, Jul. 2015.

[42] F. Caglar, S. Shekhar, and A. Gokhale, “iTune: Engineering the Performance of Xen Hypervisor via Autonomous and Dynamic Scheduler Reconfiguration,” *IEEE Transactions on Services Computing (TSC)*, vol. PP, no. 99, Mar. 2016.

[43] F. Caglar and A. Gokhale, “iOverbook: Managing Cloud-based Soft Real-time Applications in a Resource-Overbooked Data Center,” in *The 7th IEEE International Conference on Cloud Computing (CLOUD’ 14)*. Anchorage, AL, USA: IEEE, Jun. 2014, p. 10.

[44] J. J. More, “The Levenberg-Marquardt Algorithm: Implementation and Theory,” in *Numerical Analysis*, ser. Lecture Notes in Mathematics, G. Watson, Ed. Springer Berlin Heidelberg, 1978, vol. 630, pp. 105–116. [Online]. Available: http://dx.doi.org/10.1007/BFb0067700

[45] LXC. (2014) Linux Container. Last accessed: 10/11/2014. [Online]. Available: https://linuxcontainers.org/

[46] S. Shekhar, M. Walker, H. Abdelaziz, F. Caglar, A. Gokhale, and X. Koutsoukos, “A Simulation-as-a-Service Cloud Middleware,” *Journal of the Annals of Telecommunications*, vol. 74, no. 3-4, pp. 93–108, 2016.

[47] T. Dillon, C. Wu, and E. Chang, “Cloud computing: issues and challenges,” in *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*. Ieee, 2010, pp. 27–33.

[48] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, “Fog computing and its role in the internet of things,” in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*. ACM, 2012, pp. 13–16.

[49] Y. Badr, S. Hariri, Y. AL-Nashif, and E. Blasch, “Resilient and trustworthy dynamic data-driven application systems (dddas) services for crisis management environments,” *Procedia Computer Science*, vol. 51, pp. 2623 – 2637, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050915011783

[50] J. Mars, L. Tang, R. Hundt, K. Skadron, and M. L. Soffa, “Bubble-up: Increasing utilization in modern warehouse scale computers via sensible co-locations,” in *44th annual IEEE/ACM International Symposium on Microarchitecture*. ACM, 2011, pp. 248–259.

[51] S. Shekhar, A. Chhokra, A. Bhattacharjee, G. Aupy, and A. Gokhale, “INDICES: Exploiting Edge Resources for Performance-aware Cloud-hosted Services,” in *1st IEEE International Conference on Fog and Edge Computing (ICFEC) (to appear)*. Madrid, Spain: IEEE, May 2017.

[52] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, “The Case for VM-Based Cloudlets in Mobile Computing,” *Pervasive Computing, IEEE*, vol. 8, no. 4, pp. 14–23, 2009.

[53] C. Delimitrou and C. Kozyrakis, “Paragon: Qos-aware scheduling for heterogeneous datacenters,” in *ACM SIGPLAN Notices*, vol. 48, no. 4. ACM, 2013, pp. 77–88.

[54] F. Caglar, S. Shekhar, A. Gokhale, and X. Koutsoukos, “An Intelligent, Performance Interference-aware Resource Management Scheme for IoT Cloud Backends,” in *1st IEEE International Conference on Internet-of-Things: Design and Implementation*. Berlin, Germany: IEEE, Apr. 2016, pp. 95–105.

[55] K. An, F. Caglar, S. Shekhar, and A. Gokhale, “Automated Placement of Virtual Machine Replicas to Support Reliable Distributed Real-time and Embedded Systems in the Cloud,” in *International Workshop on Real-time and Distributed Computing in Emerging Applications (REACTION), 33rd IEEE Real-time Systems Symposium (RTSS ’12)*. San Juan, Puerto Rico, USA: IEEE, Dec. 2012.

[56] K. A. Faruk Caglar, Shashank Shekhar and A. Gokhale, “Intelligent Power- and Performance-aware Tradeoffs for Multicore Servers in Cloud Data Centers,” in *Proceedings of the Work-in-Progress Session of the 4th ACM/IEEE International Conference on Cyber Physical Systems (ICCPS’ 13)*. Philadelphia, PA, USA: IEEE/ACM, Apr. 2013.

[57] F. Caglar, K. An, S. Shekhar, and A. Gokhale, “Model-driven Performance Estimation, Deployment, and Resource Management for Cloud-hosted Services,” in *13th Workshop on Domain-specific Modeling (DSM ’13), In conjunction with SPLASH ’13*. Indianapolis, IN, USA: ACM, Oct. 2013.

[58] F. Caglar, S. Shekhar, and A. Gokhale, “iPlace: An Intelligent and Tunable Power- and Performance-Aware Virtual Machine Placement Technique for Cloud-based Real-time Applications,” in *17th IEEE Computer Society Symposium on Object/component/service-oriented real-time distributed Computing Technology (ISORC ’14)*. Reno, NV, USA: IEEE, Jun. 2014.

[59] ——, “Performance Interference-aware Virtual Machine Placement Strategy for Supporting Soft Real-time Applications in the Cloud,” in *3rd International Workshop on Real-time and Distributed Computing in Emerging Applications (REACTION), IEEE RTSS 2014*. Rome, Italy: IEEE, Dec. 2014, p. 6.